Lesson 1: Relational Databases & SQL

December 22, 2016

Audience

- Those who are new to relational DBs and SQL.

Objective

To understand the ***reason for relational DBs***.

To learn about the ***anatomy of relational databases***.

Understand ***relational DB relationships***.

Understand the ***importance of good database design***.

Software & Resources Needed

- Database Server with Interface
  - MS SQL Server with SQL Server Management Studio

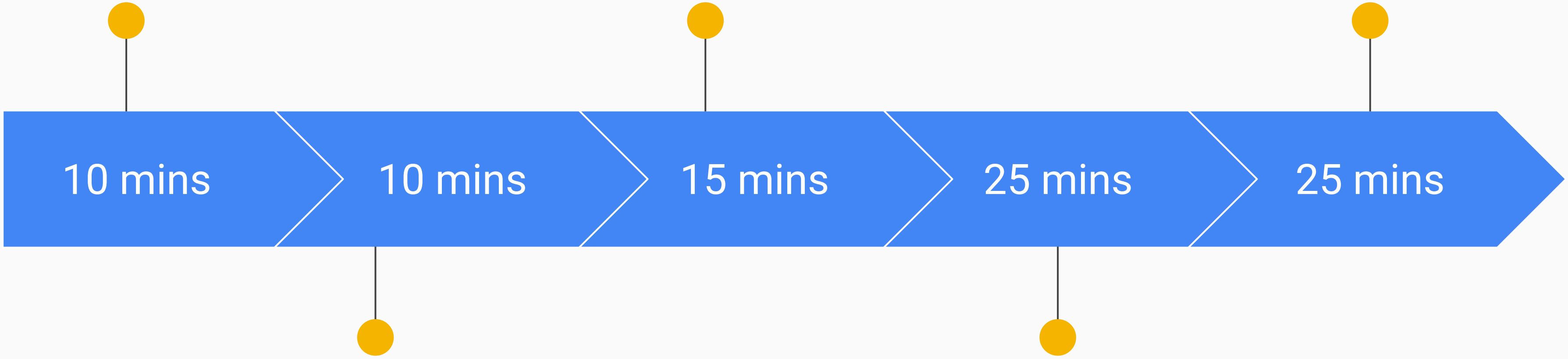
  OR

  - MySQL

  OR

  - Online (sqlfiddle.com)
- Data
  - Test data on Github repo.
- Github repo for information

Overview of Day 1



Operational vs Analytical Databases

10 mins

Why Relational Databases?

10 mins

Relational DB Anatomy

15 mins

Relationships (one-to-one, one-to-many, many-to-many)

25 mins

Good database design

25 mins

Operational vs Analytical DBs

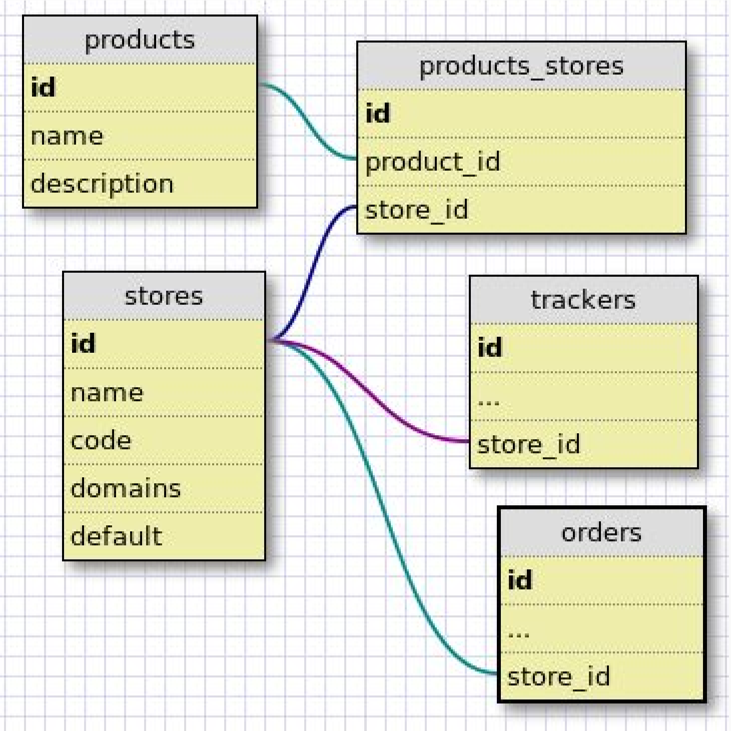
Operational DBs

Also called Transactional DBs.

Relational DBs are part of this category.

- Lots of CRUD operations performed
- Think e-commerce store. Best for dynamic, living data.

http://datawarehouse4u.info/OLTP-vs-OLAP.html



products
id
name
description

products_stores
id
product_id
store_id

stores
id
name
code
domains
default

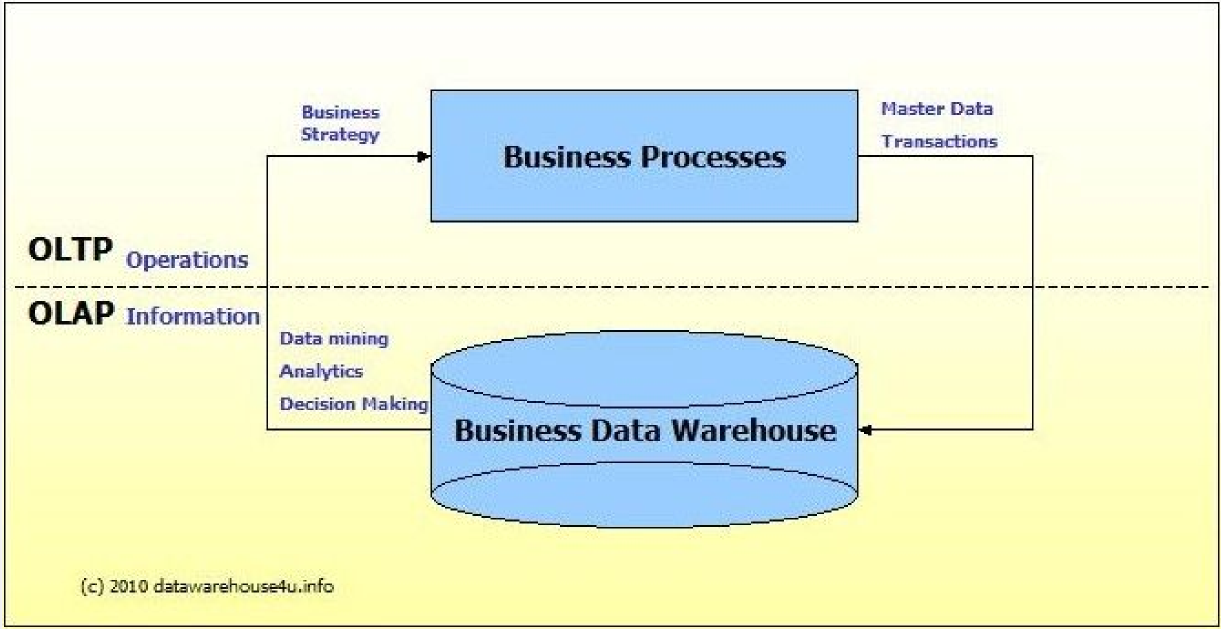
trackers
id
...
store_id

orders
id
...
store_id

Analytical DBs

- Think historical information with lots of selects.
- Best for when you report a lot and insert once in awhile, but not for data that's altered more than a rare instance.
- Great for reporting.

http://datawarehouse4u.info/OLTP-vs-OLAP.html



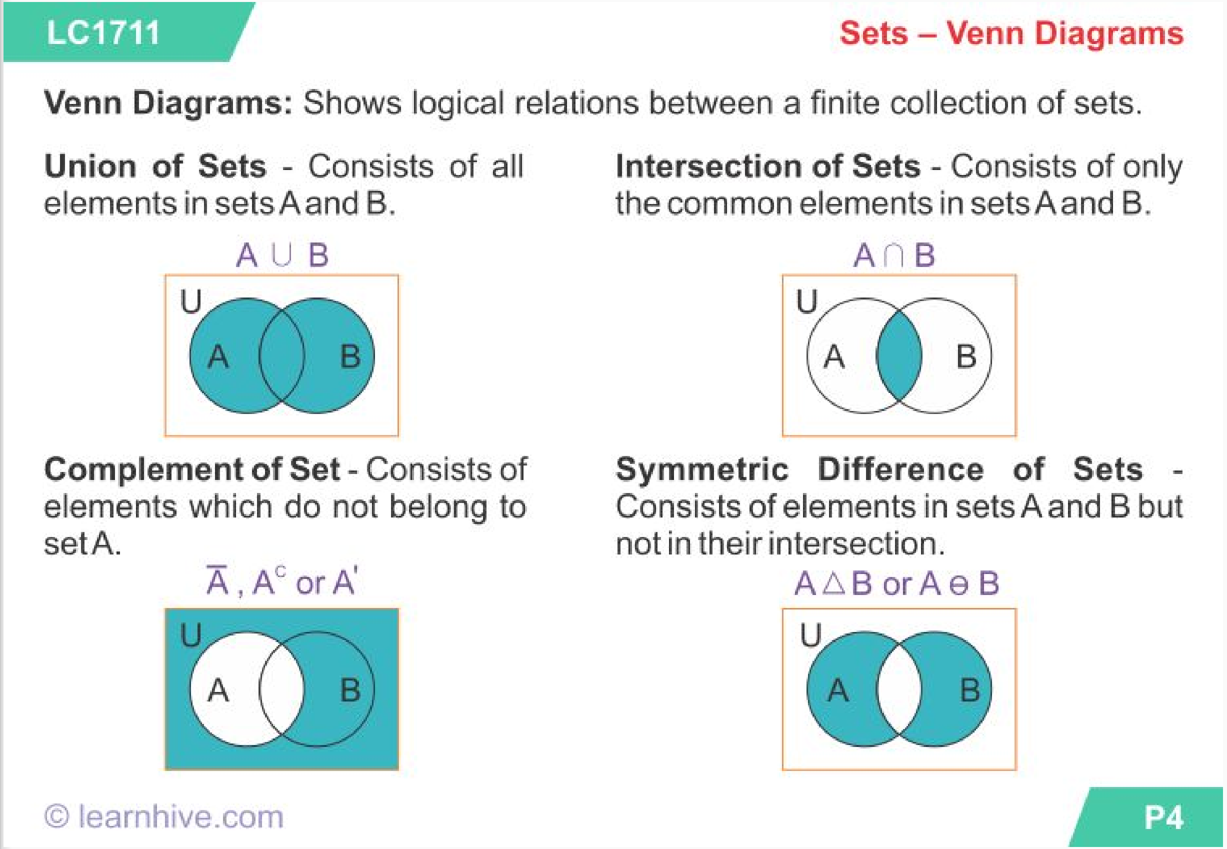
(c) 2010 datawarehouse4u.info

Relational DBs

Relational DBs

Relational DBs are operational (transactional).

- Conceived in 1969 by IBM mathematician and scientist Dr. Edgar Codd.
- He wanted a better way to store data that would keep its integrity in check.
- Looked for mathematical approach and found it.
    - Based on set theory and first-order predicate logic.



LC1711

Sets – Venn Diagrams

Venn Diagrams: Shows logical relations between a finite collection of sets.

Union of Sets - Consists of all elements in sets A and B.

$A \cup B$

Intersection of Sets - Consists of only the common elements in sets A and B.

$A \cap B$

Complement of Set - Consists of elements which do not belong to set A.

$\bar{A}$, A^c or A'

Symmetric Difference of Sets - Consists of elements in sets A and B but not in their intersection.

$A \triangle B$ or $A \ominus B$

© learnhive.com

P4

Relational Database Systems

- Relational Database Management Systems (RDBMS) are software programs to work with relational databases.
    - Microsoft SQL Server
    - PostgreSQL
    - MySQL
    - IBM DB2
    - Oracle
    - . . .

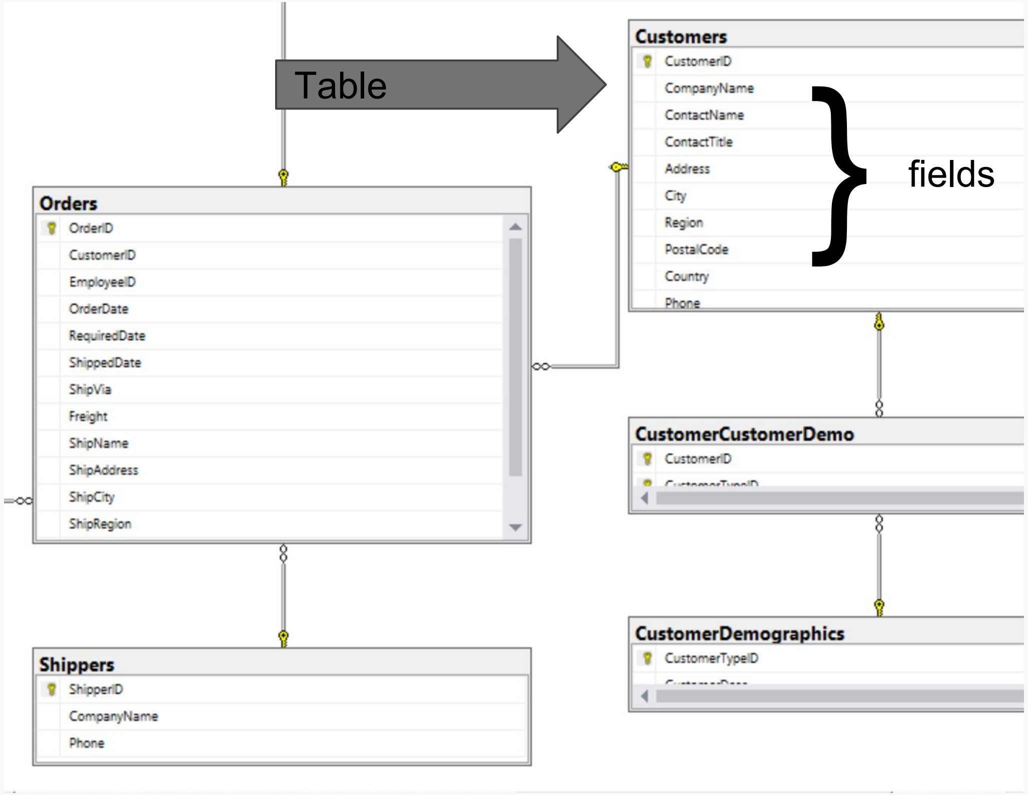
Anatomy of Relational DBs

Anatomy of Relational DBs Overview

- Tables
- Fields
- Records
- Keys
- Views
- Relationships

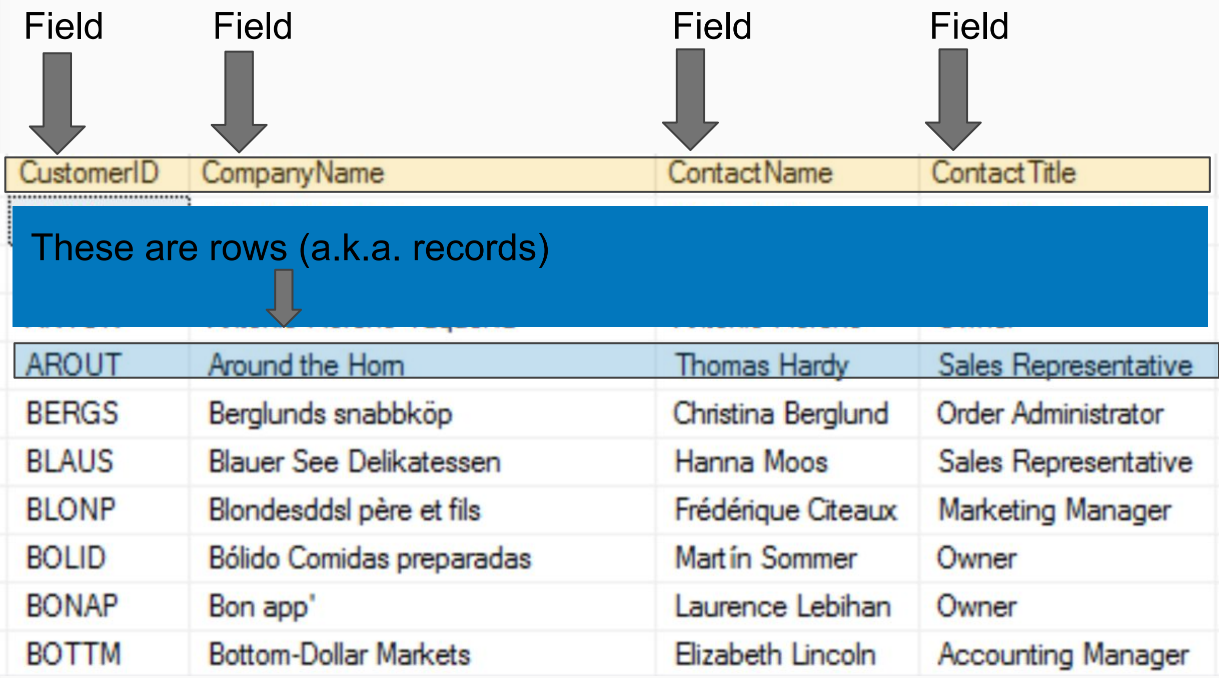
Tables

- Main structures in db
- Has at least 1 field
    - Primary Key
        - Unique Identifier
- Represents *single* specific subject
    - Object
        - E.g. “Customers”
    - Event
        - E.g. “Customer Orders”



Fields

- Smallest structure in a database.
- Represents a characteristic of the table’s subject.
  - Like a “property” in OO.
- Contains ONE and only ONE value
  - Color : Brown NOT ~~Color: Brown, Red, Green~~



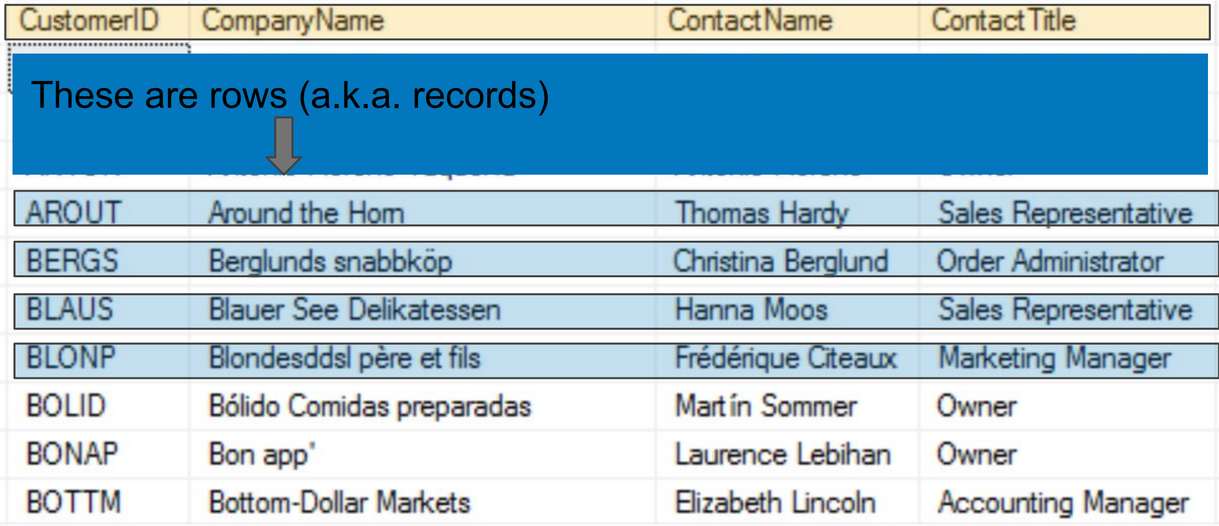
Field Field Field Field

These are rows (a.k.a. records)

CustomerID	CompanyName	ContactName	ContactTitle
AROUT	Around the Hom	Thomas Hardy	Sales Representative
BERGS	Berglunds snabbköp	Christina Berglund	Order Administrator
BLAUS	Blauer See Delikatessen	Hanna Moos	Sales Representative
BLONP	Blondesddsl père et fils	Frédérique Citeaux	Marketing Manager
BOLID	Bólido Comidas preparadas	Martín Sommer	Owner
BONAP	Bon app'	Laurence Lebihan	Owner
BOTTM	Bottom-Dollar Markets	Elizabeth Lincoln	Accounting Manager

Records

- Represents a unique instance of the subject of a table
    - Think of these as an object instance, whereas the table would be like a class.
- Composed of the entire set of fields in a table (even if some fields in that row have null or empty values).
- On the right, each record represents a single customer.

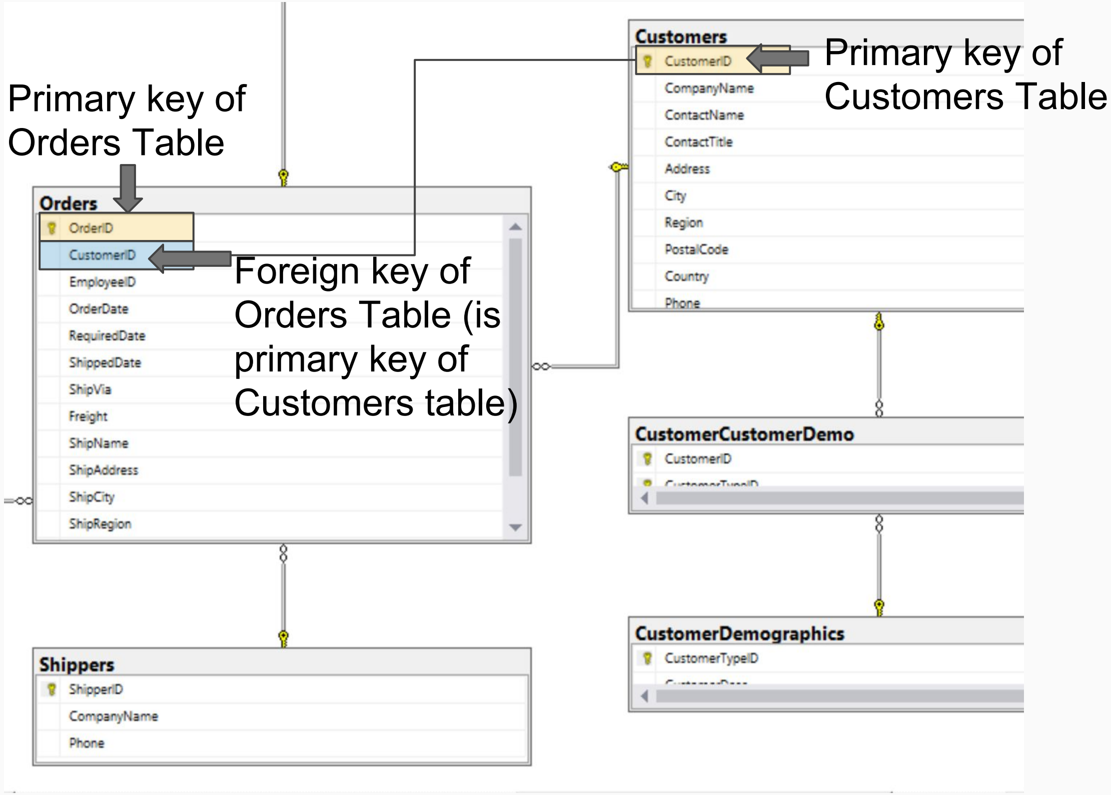


These are rows (a.k.a. records)

CustomerID	CompanyName	ContactName	ContactTitle
AROUT	Around the Horn	Thomas Hardy	Sales Representative
BERGS	Berglunds snabbköp	Christina Berglund	Order Administrator
BLAUS	Blauer See Delikatessen	Hanna Moos	Sales Representative
BLONP	Blondesddsl père et fils	Frédérique Citeaux	Marketing Manager
BOLID	Bólido Comidas preparadas	Martín Sommer	Owner
BONAP	Bon app'	Laurence Lebihan	Owner
BOTTM	Bottom-Dollar Markets	Elizabeth Lincoln	Accounting Manager

Keys

- Special fields within a table.
- Different types of Keys, but 2 most used are:
  - Primary keys
    - Unique identifier for a record in the table.
    - Like US citizens have Social Security number.
  - Foreign keys
    - Helps establish relationships between tables.
    - Ensures integrity.
    - It’s a primary key from one table that resides in another.



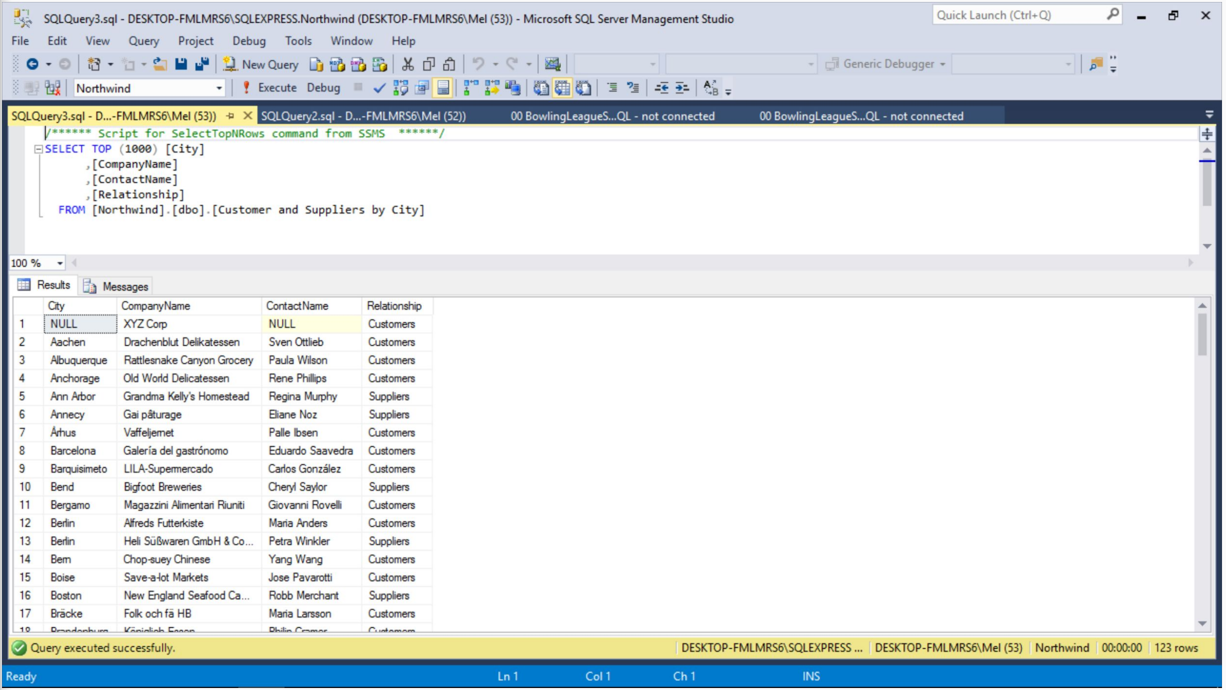
Primary key of Orders Table

Primary key of Customers Table

Foreign key of Orders Table (is primary key of Customers table)

Views

- A virtual table made up of fields from other tables (one or more)
- Let’s you see info from your database in other ways (other views).



SQLQuery3.sql - DESKTOP-FMLMRS6\SQLEXPRESS.Northwind (DESKTOP-FMLMRS6\Mel (53)) - Microsoft SQL Server Management Studio
Quick Launch (Ctrl+Q)
File Edit View Query Project Debug Tools Window Help
New Query
Generic Debugger
Northwind
Execute Debug
SQLQuery3.sql - D...-FMLMRS6\Mel (53))
SQLQuery2.sql - D...-FMLMRS6\Mel (52))
00 BowlingLeagueS...QL - not connected
00 BowlingLeagueS...QL - not connected
100 %
Results
Messages

```
/****** Script for SelectTopNRows command from SSMS  ******/
SELECT TOP (1000) [City]
      ,[CompanyName]
      ,[ContactName]
      ,[Relationship]
  FROM [Northwind].[dbo].[Customer and Suppliers by City]
```

	City	CompanyName	ContactName	Relationship
1	NULL	XYZ Corp	NULL	Customers
2	Aachen	Drachenblut Delikatessen	Sven Ottlieb	Customers
3	Albuquerque	Rattlesnake Canyon Grocery	Paula Wilson	Customers
4	Anchorage	Old World Delicatessen	Rene Phillips	Customers
5	Ann Arbor	Grandma Kelly's Homestead	Regina Murphy	Suppliers
6	Annecy	Gai pâturage	Eliane Noz	Suppliers
7	Århus	Vaffeljernet	Palle Ibsen	Customers
8	Barcelona	Galería del gastrónomo	Eduardo Saavedra	Customers
9	Barquisimeto	LILA-Supermercado	Carlos González	Customers
10	Bend	Bigfoot Breweries	Cheryl Saylor	Suppliers
11	Bergamo	Magazzini Alimentari Riuniti	Giovanni Rovelli	Customers
12	Berlin	Alfreds Futterkiste	Maria Anders	Customers
13	Berlin	Heli Süßwaren GmbH & Co...	Petra Winkler	Suppliers
14	Bern	Chop-suey Chinese	Yang Wang	Customers
15	Boise	Save-a-lot Markets	Jose Pavarotti	Customers
16	Boston	New England Seafood Ca...	Robb Merchant	Suppliers
17	Bräcke	Folk och fä HB	Maria Larsson	Customers
18	Brandenburg	Königlich Essen	Philip Cramer	Customers

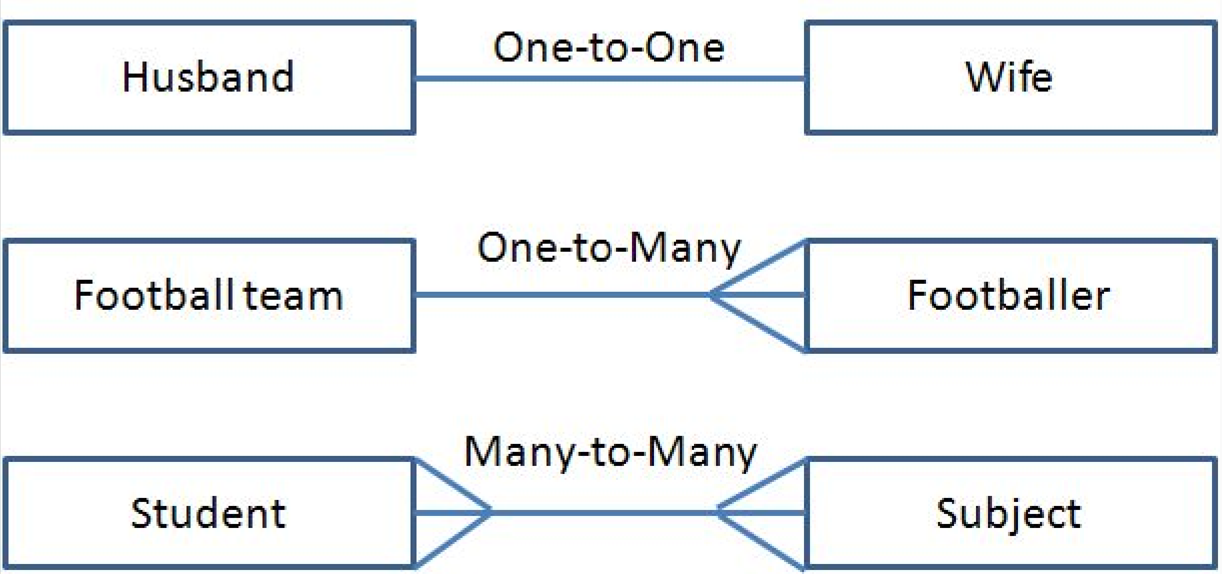
Query executed successfully.

DESKTOP-FMLMRS6\SQLEXPRESS ... | DESKTOP-FMLMRS6\Mel (53) | Northwind | 00:00:00 | 123 rows

Ready   Ln 1   Col 1   Ch 1   INS

Relationships

- How tables are linked or joined together.
- 3 Types:
  - One-to-One
  - One-to-Many
  - Many-to-Many



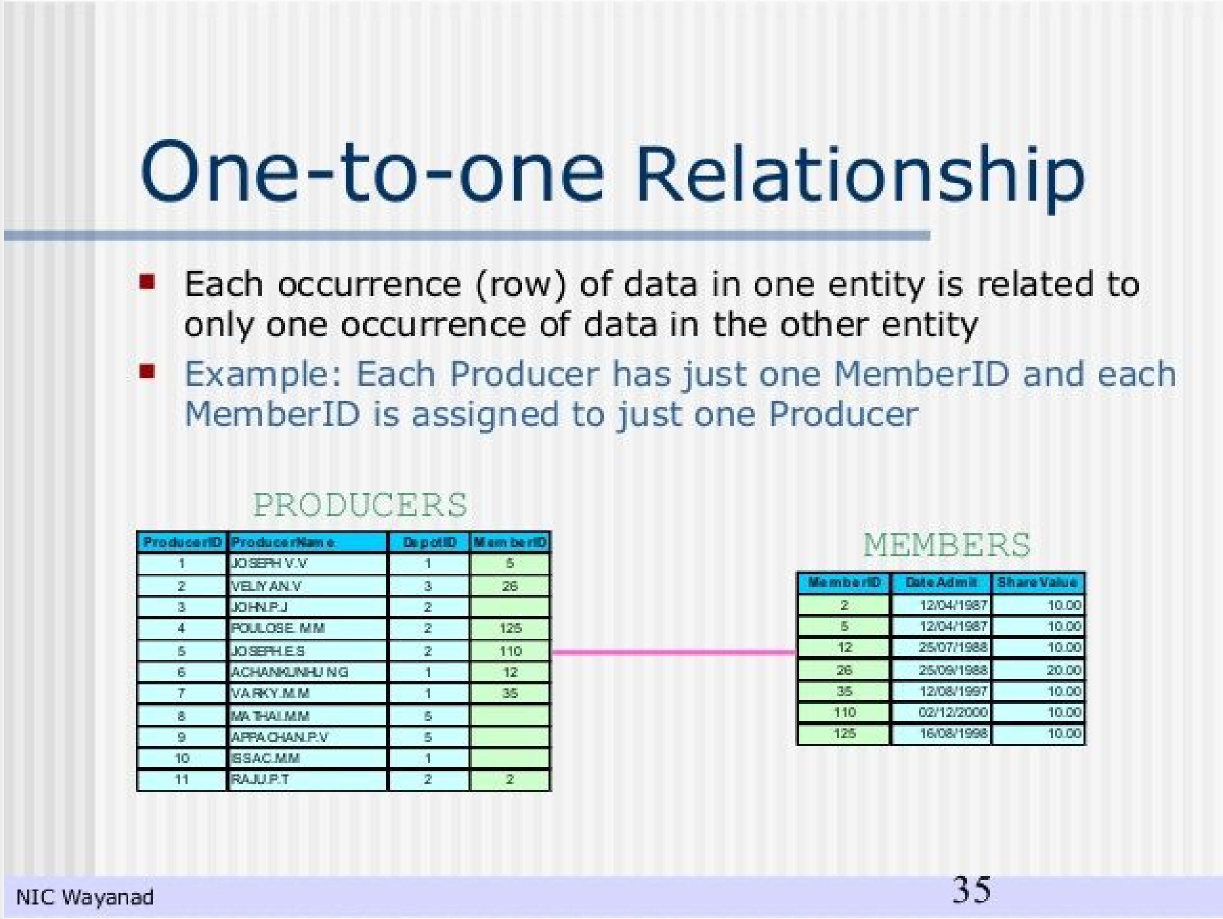
Types of Relationships

Types of Relationships Overview

- One-to-one
- One-to-many
- Many-to-many

One-to-One

- One record from the first table is related to one and only one record of the second table.
- Rare -- because these could really be in a single table (and often are)
- Use cases :
    - When security is important for certain fields.
    - When you’re pulling data in from another source.



One-to-one Relationship

- Each occurrence (row) of data in one entity is related to only one occurrence of data in the other entity
- Example: Each Producer has just one MemberID and each MemberID is assigned to just one Producer

PRODUCERS

ProducerID	ProducerName	DepotID	MemberID
1	JOSEPH V.V	1	5
2	VELIYAN.V	3	26
3	JOHN.P.J	2	
4	POULOSE. M.M	2	125
5	JOSEPH.E.S	2	110
6	ACHANKUNHU N.G	1	12
7	VARKY.M.M	1	35
8	MATHAI.M.M	5	
9	APPACHAN.P.V	5	
10	ISSAC.M.M	1	
11	RAJU.P.T	2	2

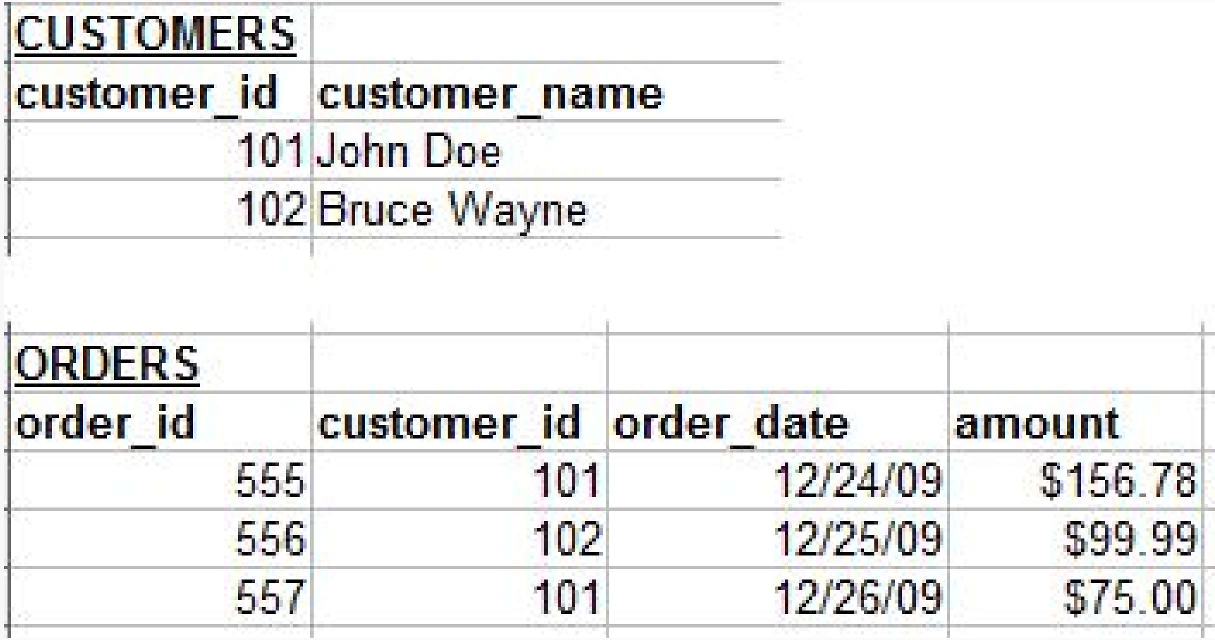
MEMBERS

MemberID	Date Admit	Share Value
2	12/04/1987	10.00
5	12/04/1987	10.00
12	25/07/1988	10.00
26	25/09/1988	20.00
35	12/08/1997	10.00
110	02/12/2000	10.00
125	16/08/1998	10.00

NIC Wayanad 35

One-to-Many

- A single record from the first table can be related to many records from the second table, BUT a record from the second table can only be related to one record from the first table.
- Very common scenario.
- Example: One customer can have many orders, but a single order can only have one customer.



CUSTOMERS

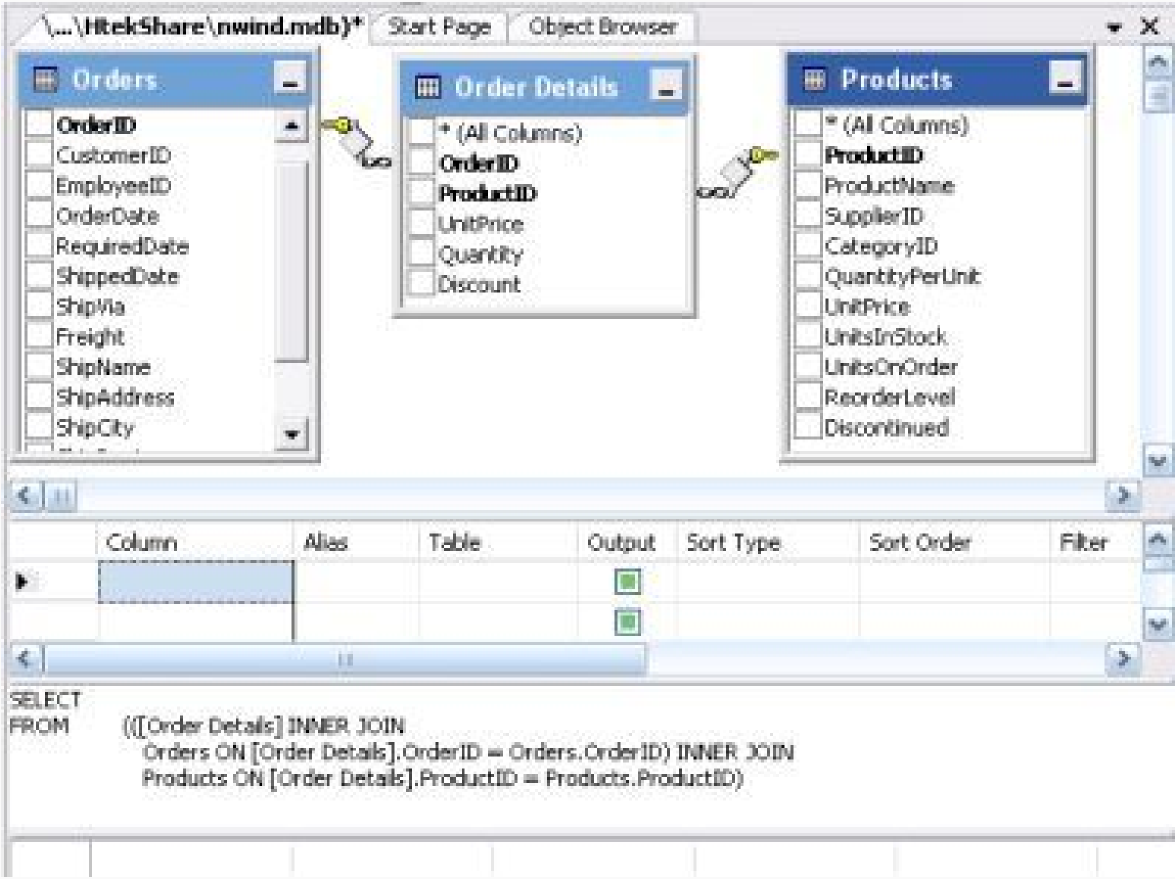
customer_id	customer_name
101	John Doe
102	Bruce Wayne

ORDERS

order_id	customer_id	order_date	amount
555	101	12/24/09	$156.78
556	102	12/25/09	$99.99
557	101	12/26/09	$75.00

Many-to-Many

- A single record from the first table can be related to many records from the second table, AND a single record from the second table can be related to many records from the first table.
- Very common.
- Example: One product can be in many orders, and many orders can have the same product.
- Note: You can’t do this in 2 tables (try it yourself). You need a third joining or linking table in this scenario.



```
SELECT
FROM        (([Order Details] INNER JOIN
              Orders ON [Order Details].OrderID = Orders.OrderID) INNER JOIN
              Products ON [Order Details].ProductID = Products.ProductID)
```

Sound DB Structure

Sound DB Structure Overview

- Importance of sound DB architecture
- Good naming
- No duplicate fields
- Primary key
- Appropriate foreign keys
- No calculated values
- No multipart fields
- No multivalued fields

Importance of sound DB structure

- Without a properly designed database, SQL queries won’t be easy, and will often be incorrect.
- This is often left to a DBA (database administrator), but it’s important to understand how to spot a poorly designed db so you know what you’re dealing with.

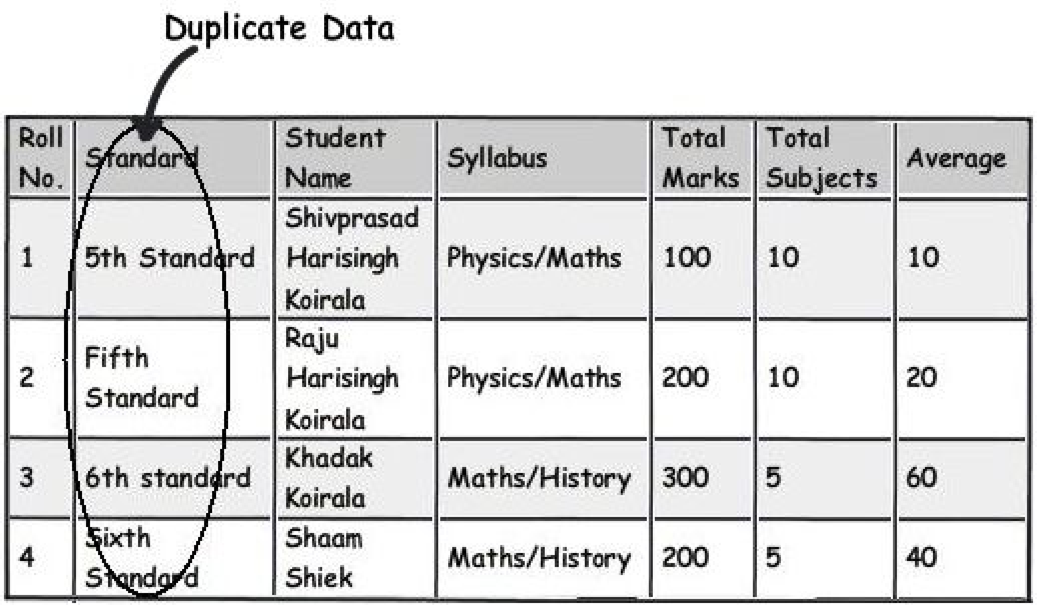
Good Naming

- Be thoughtful when naming your database, tables, fields, and views
  - Be specific
  - Think about your organization’s conventions
  - Examples
    - Good: CustomerCellPhoneNumber
    - Bad: PhoneNumber

NO Duplicate Fields

- If you see the same field throughout tables in the database, you have a problem. DON'T DO THIS.
- Hurts the integrity of data. What if you change it one place, but not the other?

Wrong



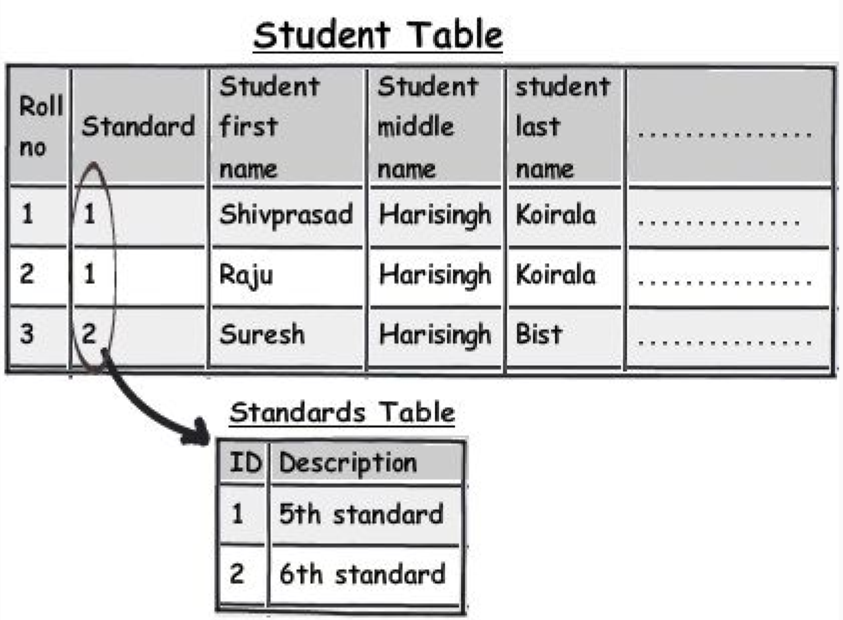
Duplicate Data

Roll No.	Standard	Student Name	Syllabus	Total Marks	Total Subjects	Average
1	5th Standard	Shivprasad Harisingh Koirala	Physics/Maths	100	10	10
2	Fifth Standard	Raju Harisingh Koirala	Physics/Maths	200	10	20
3	6th standard	Khadak Koirala	Maths/History	300	5	60
4	Sixth Standard	Shaam Shiek	Maths/History	200	5	40

NO Duplicate Fields

- Instead, use a referential table and keys.

Right



Student Table

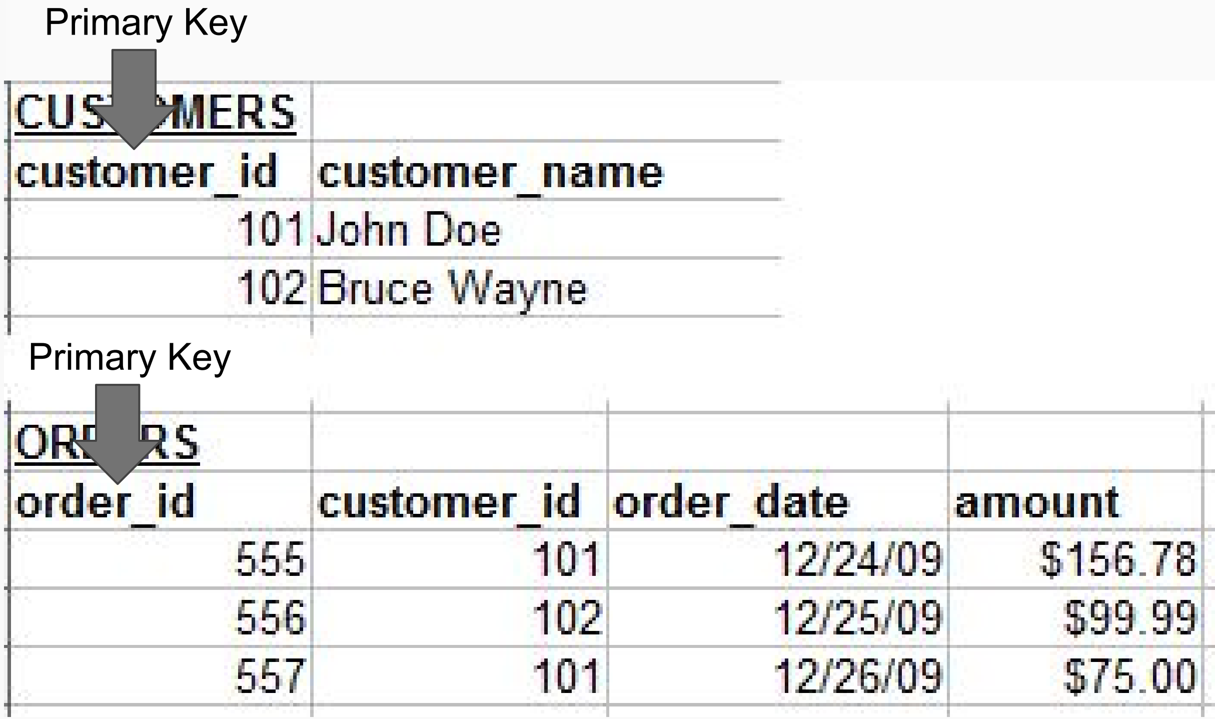
Roll no	Standard	Student first name	Student middle name	student last name	....
1	1	Shivprasad	Harisingh	Koirala	....
2	1	Raju	Harisingh	Koirala	....
3	2	Suresh	Harisingh	Bist	....

Standards Table

ID	Description
1	5th standard
2	6th standard

Primary Key

- Each table should have a distinct identifier.
- You use this as a handle in order to know you are grabbing the right records.
- Typically a primary key is a single field, but in cases of joining (linking) tables, you'll sometimes see 2 foreign keys become a composite primary key for that table.



Primary Key

CUSTOMERS

customer_id	customer_name
101	John Doe
102	Bruce Wayne

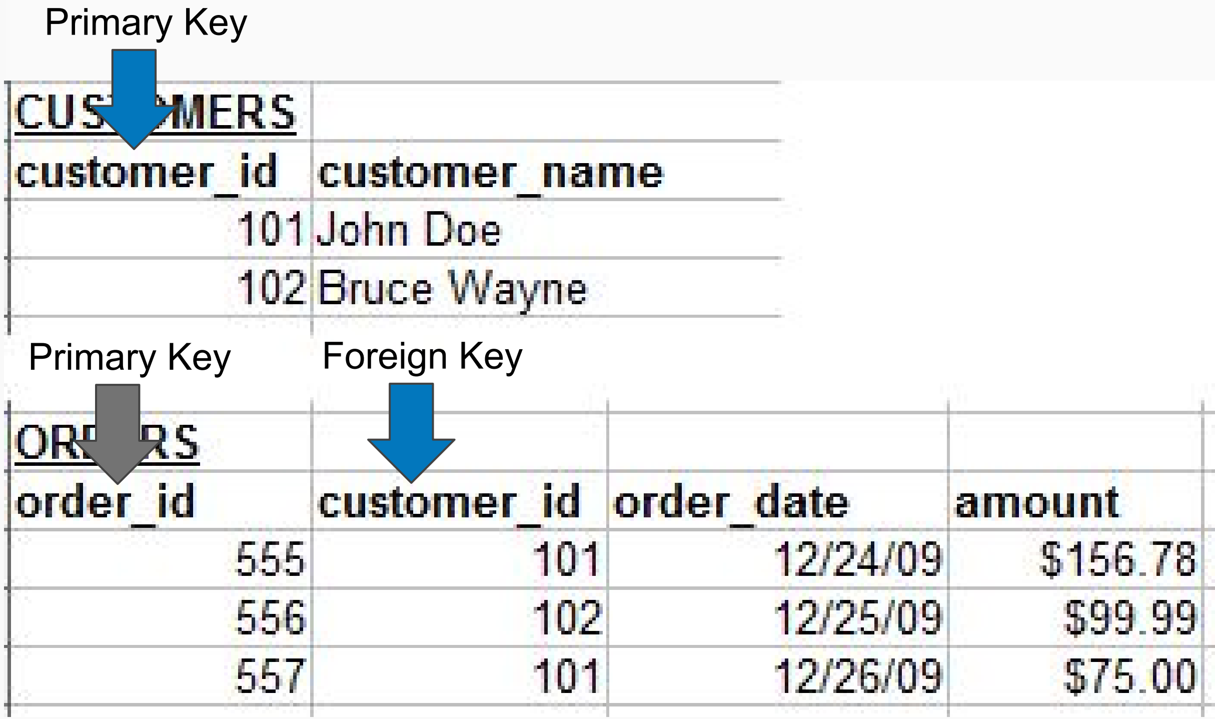
Primary Key

ORDERS

order_id	customer_id	order_date	amount
555	101	12/24/09	$156.78
556	102	12/25/09	$99.99
557	101	12/26/09	$75.00

Foreign Key

- In order to create one-to-many and many-to-many relationships between tables, you'll use foreign keys.
- This will also help maintain the integrity of the data.
- You can have multiple foreign keys in a table, especially in joining (linking) tables.
- The example on the right shows a one-to-many relationship between customers and orders.


Primary Key
Primary Key
Foreign Key

CUSTOMERS

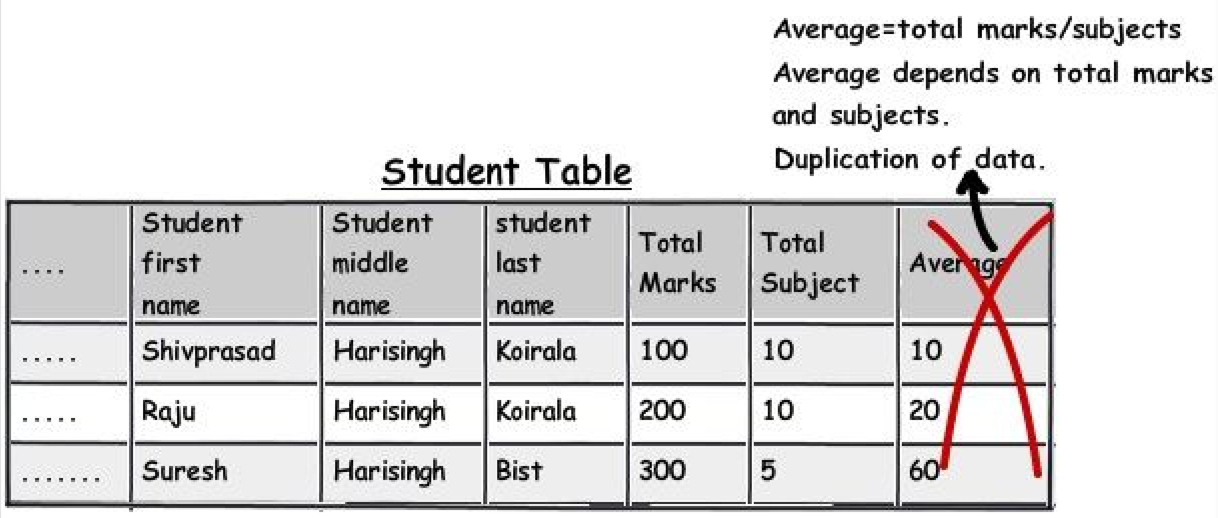
customer_id	customer_name
101	John Doe
102	Bruce Wayne

ORDERS

order_id	customer_id	order_date	amount
555	101	12/24/09	$156.78
556	102	12/25/09	$99.99
557	101	12/26/09	$75.00

NO Calculated Fields

- No field should depend upon another, non-primary key field.
- The field called Average (right) depends on the 2 non-primary fields, Total Marks and Total Subject.
- This data duplication can lead to all sorts of issues. Think about if Total Subject is updated, for example. The Average would change. If it isn't updated, the integrity of the data is in jeopardy.



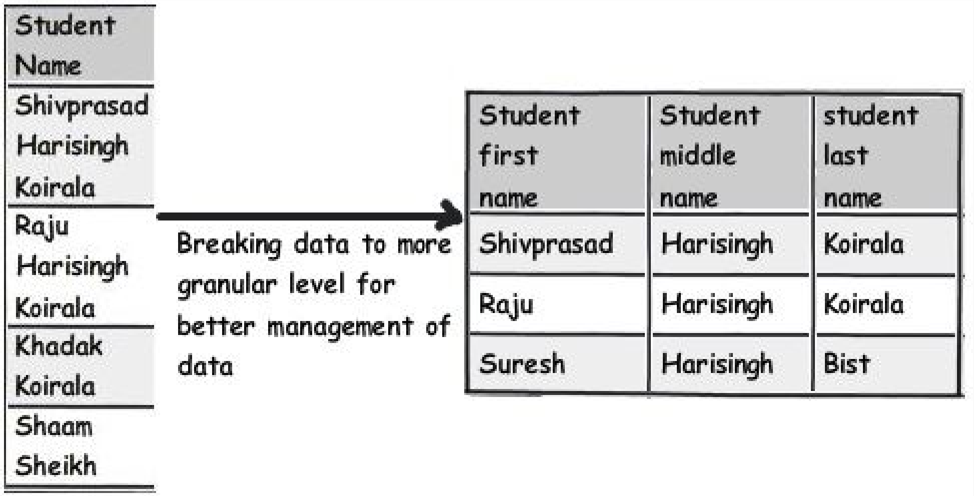
Average=total marks/subjects
Average depends on total marks and subjects.
Duplication of data.

Student Table

....	Student first name	Student middle name	student last name	Total Marks	Total Subject	Average
.....	Shivprasad	Harisingh	Koirala	100	10	10
.....	Raju	Harisingh	Koirala	200	10	20
....	Suresh	Harisingh	Bist	300	5	60

NO Multipart Fields

- Keep information as granular as possible (without going overboard).
- Student name (right) will cause issues when searching and sorting.
    - Sort by first or last name is difficult if they are combined in a single field.



Student Name
Shivprasad Harisingh Koirala
Raju Harisingh Koirala
Khadak Koirala
Shaam Sheikh

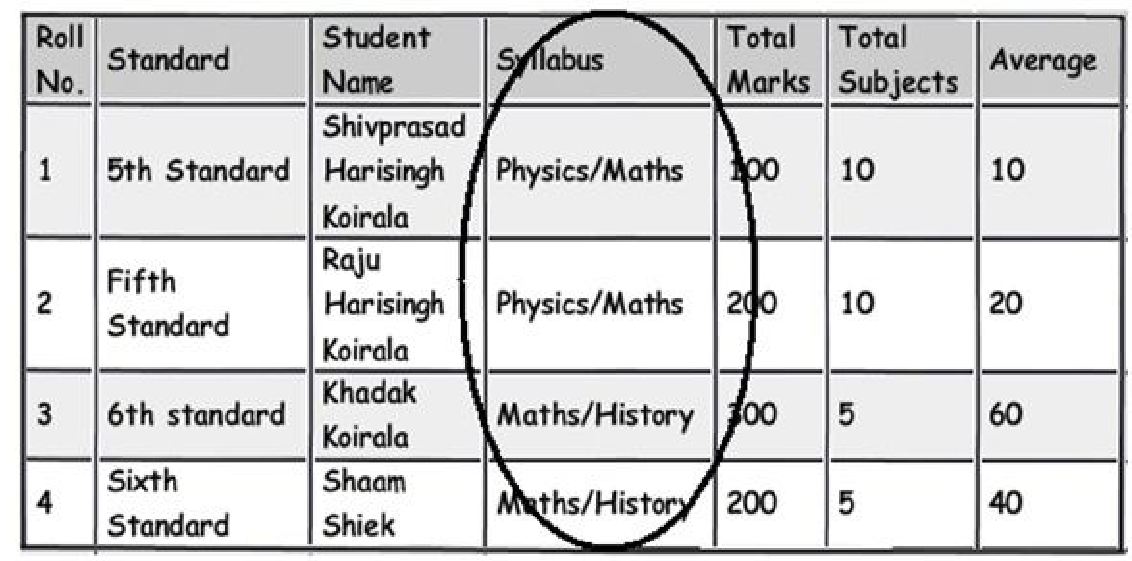
Breaking data to more granular level for better management of data

Student first name	Student middle name	student last name
Shivprasad	Harisingh	Koirala
Raju	Harisingh	Koirala
Suresh	Harisingh	Bist

NO Multivalue Fields

- Avoid repeating groups.
- Imagine if someone insert incorrectly. Imagine trying to search for all Maths. It becomes very difficult.
- Data integrity threatened.
- Instead, break this data out and create table relationships (shown on next slide).

Wrong

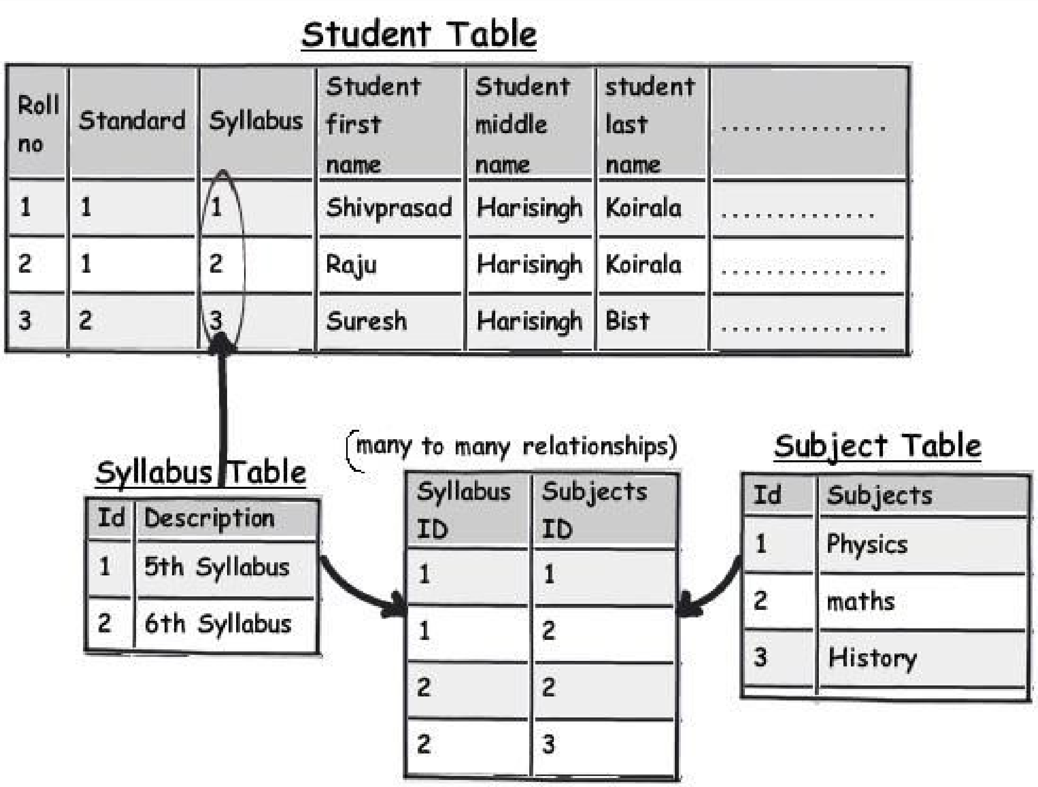


Roll No.	Standard	Student Name	Syllabus	Total Marks	Total Subjects	Average
1	5th Standard	Shivprasad Harisingh Koirala	Physics/Maths	100	10	10
2	Fifth Standard	Raju Harisingh Koirala	Physics/Maths	200	10	20
3	6th standard	Khadak Koirala	Maths/History	300	5	60
4	Sixth Standard	Shaam Shiek	Maths/History	200	5	40

NO Multivalue Fields

- Ensure you break down data appropriately and don't stuff a bunch of data into a single field.

Right



Student Table

Roll no	Standard	Syllabus	Student first name	Student middle name	student last name	....
1	1	1	Shivprasad	Harisingh	Koirala	....
2	1	2	Raju	Harisingh	Koirala	....
3	2	3	Suresh	Harisingh	Bist	....

Syllabus Table

Id	Description
1	5th Syllabus
2	6th Syllabus

(many to many relationships)

Syllabus ID	Subjects ID
1	1
1	2
2	2
2	3

Subject Table

Id	Subjects
1	Physics
2	maths
3	History

Homework

Ensure a SQL Server is installed.

MS SQL Server with SQL Server Management Studio

OR

MySQL

Ensure you have run the download scripts to create and populate your databases. Script located in zip file on the Github repository.

https://github.com/MelMcGee/sql-in-7/blob/master/SQLQFMM3.zip

View the Readme file in the zipped folder for instructions.